Networked control and observation for Master-Slave systems

Wenjuan Jiang, Alexandre Kruszewski, Jean-Pierre Richard, Armand Toguyeni

Networked control and observation for Master-Slave systems

Wenjuan Jiang[1],Alexandre Kruszewski[1], Jean-Pierre Richard[1,2] and Armand Toguyeni[1]

[1] LAGIS CNRS UMR 8146, Ecole Centrale de Lille, BP 48, 59651 Villeneuve d'Ascq Cedex, France.
wenjuan.jiang@ec-lille.fr, Alexandre.Kruszewski@ec-lille.fr, jean-pierre.richard@ec-lille.fr and armand.toguyeni@ec-lille.fr

[2] Equipe-Projet ALIEN, INRIA.

Summary. This chapter concerns the design of a remote control loop constituted by a Slave system (with computing and energy limitations) and a Master computer, communicating via an Internet connection. In such a situation, the communication cost is reduced but the Quality of Service of the Internet connection is not guaranteed. In particular, when the Slave dynamics are expected to be fast enough, the network induces perturbations (delays, *jitters*, packet dropouts and sampling) that may damage the performance. Here, the proposed solution relies on a delay-dependent, state-feedback control, computed by the Master on the basis of an observer. This last estimates the present Slave's state from its past sampled outputs, despite the various delays. Then, the computing task is concentrated in the Master. The theoretical results are based on the Lyapunov-Krasovskii functional and the approach of LMI, which guarantee the stabilization performance with respect to the expected maximum delay of the connection. Two strategies are applied: one is a constant controller/observer gain strategy, which takes into account a fixed upperbound for the communication delay. The second strategy aims at improving the performance by adapting the gains to the available network QoS (here, with two possible upperbounds).

Key words: Remote control, Switching signal, Exponential stability, Linear time-delay system, LMIs, Internet, UDP, Robot.

1 Introduction

Networked Control System (NCS) is a type of closed-loop control system with real-time communication networks imported into the control channel and feedback channel. The control and feedback signals in a NCS exchanged among the system's components are in the form of information packets through a network. The network-induced delay is brought into the control systems, which may create unstable behaviors.

Network induced delays vary depending on the network hardware, on the different protocols, on the traffic load... In some cases, such as in token ring local area network, the time-delay is bounded; for other networks like Ethernet and Internet, the time-delay is unbounded and varying.

As Internet and Ethernet are well developed, remote control system has been widely used in industrial, communicational, medical systems, to cite a few. However, alongside the advantage of low costs, the Internet inevitably brings problems to the closed-loop controlled system, such as delay variation, data-packets loss [1] and disorder, which can cause poor performance, instability or danger(see for instance chapiter 5 of [2],[3] and the references herein).

How to diminish the effect of time delay in the remote system is critical in the system design. The main solution can split into two (combinable) strategies [2, 4]: 1) Increase the network performances (QoS) or 2) design an adapted control that can compensate the network influence. In this chapter, we consider this last approach for the network controlled system via Internet or Ethernet. The experiments we propose in the last part are using Internet, but the control strategy holds for both network standards.

A variety of stability and control techniques have been developed for general time delay systems [5, 6, 7, 8]. Applications of these techniques to Networked Control Systems were also derived [9, 10, 11, 12, 1, 13, 14]. But some of these results are based on simplifying assumptions (for instance, the delays are constant) or lead to technical solutions that decrease the performances (for instance, a "buffer strategy" allows to make the communication delays become constant by waiting enough after the data are received). In fact, to consider the time delay as constant [10, 15, 16, 17] is actually unrealistic due to the dynamic character of the network. A delay maximizing strategy [8, 11] ("virtual delay", "buffer", or "waiting" strategy) can be carried out so to make the delay constant and known. This requires the knowledge of the maximum delay values h_m. However, it is obvious that maximizing the delay up to its largest value decreases the speed performance of the remote system. Several results are limited to time-delay whose value is less than the sensor and controller sampling periods [18]. In the Internet case, this constraint leads to increase the sampling periods up to the maximal network delay, which may be constraining for high dynamic applications.

Note that, in the Internet case, the network delays cannot be modeled nor predicted. Moreover, the (variable) transmission delays are asymmetric, which means that the delay $h_1(t)$ from Master to Slave (shortly, M-to-S), and the return one (S-to-M) $h_2(t)$ normally satisfy $h_1(t) \neq h_2(t)$. Because of this lack of knowledge, predictor-based control laws [12] cannot be applied.

Our aim is to ensure suitable stabilization and speed performances, *i.e.* exponential stabilization, despite the dynamic variations of the network.

Our solution relies on the theoretical results of [19] (exponential stabilization of systems with unknown, varying delays), as well as [13] (networked control), the main lines of which will be shortly recalled in the next section. It allows for applying a waiting strategy only to the M-to-S communication, whereas the S-to-M communication takes the sensor measurements into account as soon as received. In order

to enhance the performance of the system, a gain scheduling strategy is adapted according to the variable time-delay of the network involved. In our application, we use Internet for a long distance remote control. The Master-Slave system is based on the UDP protocol and involves lists as buffers. The choice of UDP is preferred to TCP because in our NCS (Networked Control System) situation, packets re-emitting is not needed and setting up the TCP connection between two PCs is time-consuming.

2 Exponential stability of a remote system controlled through Internet

We consider the remote system based on the Master-Slave structure. For energy saving reasons, the work of the Slave PC is simplified, while the control and observation complexity is concentrated on the Master. The main features of the system refer to Fig.1.

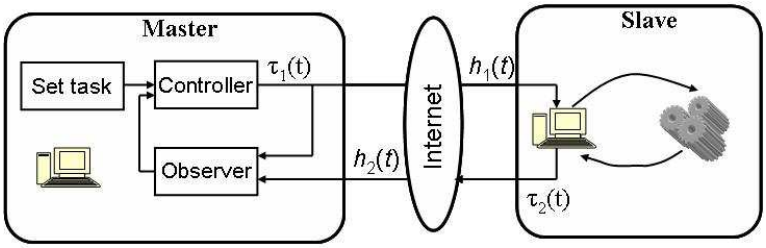

Fig. 1. Structure of general M/S based remote system

2.1 The three delay sources

In such a situation, the variable delays come from: 1) the communication through the Internet; 2) the data-sampling; and 3) the possible packet losses. In the sequel, $h_1(t)$ and $h_2(t)$ denote the communication delays and $\tau_1(t)$ and $\tau_2(t)$ include the sampling delays and possible packet losses. The total delay $\delta_i(t)$ between Master and Slave results from the addition of $h_i(t)$ and $\tau_i(t)$ for $i \in \{1,2\}$.

1) Both computers dates are automatically synchronized in the system. The strategy of NTP (Network Time Protocol)[20] is used in the program to calculate the time clock difference between the two sides. By this way, whenever the Master or the Slave receives the data including the time stamp, it knows the instant t_k of data sending out and the resulting transmission delay $h_i(t_k), i = 1,2$.

2) The real remote system, including Master, Slave and network, must involve some data sampling. However, following [21, 22], this phenomenon is equivalent to a time-varying, discontinuous delay $\tau_i(t)$ (to be defined in (1)), which allows for

keeping a continuous-time model. If the packets exchange between the Master and the Slave is of high speed, then $\tau_i(t)$ constitutes a disturbance that should be considered in the stabilization design [1]. $\tau_i(t)$ is variable but it is supposed there is a known T (maximum sampling period) so that $\tau_i(t) \leq T$.

3) If some packet p_{t_k} containing the sample at t_k is lost, or arrives later than the packet $p_{t_{k+1}}$, then the Master only considers the most recent data (i.e., those from $p_{t_{k+1}}$). If it is assumed that the maximum number of successive packets that can be lost is N, then the maximum resulting delay is NT. The same lines also holds for the control packets.

From 2) and 3), the delay $\delta_i(t)$ has a known maximum $\delta_i^m(t) = (N+1)T + h_m$ and the delay variation satisfies $\dot{\delta}_i(t) \leq 1$. In order to keep simple expressions, the notation T will be kept preferably to $T' = (N+1)T$.

Summarizing, given a signal $g(t)$ and the global delay $\delta(t)$ which represents the combination of the sampling and packet delay $h(t_k)$ that the transmission line subjects to the packet containing the k^{th} sample at time t_k, $g(t)$ can be written as:

$$\begin{array}{ll} g(t_k - h(t_k)) & = g(t - h(t_k) - (t - t_k)), \\ & = g(t - \delta(t)), \\ t_k \leq t < t_{k+1}, & \delta(t) \triangleq h(t_k) + \tau_k(t), \\ & \tau_k(t) = t - t_k. \end{array} \tag{1}$$

2.2 Transmission and receipt of the control data

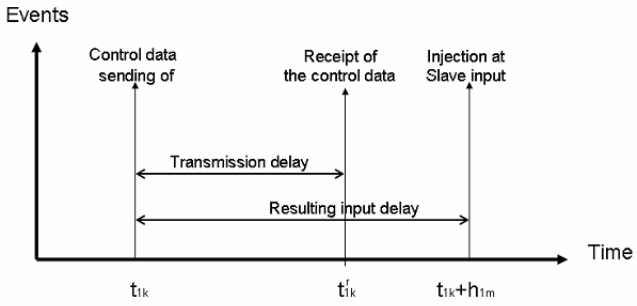

Fig. 2. Control data processing

The k^{th} data sent by the Master to Slave includes the control $u(t_{1,k})$ together with the time stamp when the packet is sent out. At time $t^r_{1,k}$, when the Slave receives the data it can calculate the time delay because of the time stamp. If the delay equals h_{1m}, the Slave should apply immediately the command.

The control u, sent out by the Master at time $t_{1,k}$, is received by the Slave at time $t^r_{1,k} > t_{1,k}$. It will be injected in the Slave input only at the pre-defined “target time"

$t_{1,k}^{target} = t_{1,k} + h_{1m}$. The corresponding waiting time $t_{1,k} + h_{1m} - t_{1,k}^{r}$ is depicted on Fig. 2. This is realistic because the transmission delay is bounded by a known value h_{1m}. By this way, the Master knows the time $t_{1,k} + h_{1m}$ when this control $u(t_{1,k})$ will be injected at the Slave input.

2.3 Problem formulation and preliminaries

Consider the Slave as a linear system. It is described as following form, in which (A,B,C) is controllable and observable.

$$\begin{cases} \dot{x}(t) = Ax(t) + Bu(t - \delta_1(t)), \\ y(t) = Cx(t), \end{cases} \tag{2}$$

where $\delta_1(t) = \delta_1 + \eta_1(t), \| \eta_1(t) \| \leq \mu_1$.

In order to guarantee the closed-loop stability whatever the delay variation, the exponential stability with the rate α must be achieved. In other words, there must be a real $\kappa \geq 1$ so that the solution $x(t;t_0,\phi)$ starting at any time t_0 from any initial function ϕ satisfies:$\|x(t;t_0,\phi)\| \leq \kappa \|\phi\| e^{-\alpha(t-t_0)}$. In this paper, it is achieved using a state observer and a state feedback.

To ensure this exponential global stabilization, one can use the results of [13] which considers a Lyapunov-Krasovskii functional with descriptor representation:

$$\begin{array}{l} V(t) = \bar{x}_\alpha^T(t) E P \bar{x}_\alpha(t) + \int_{-\delta_1}^{0} \int_{t+\theta}^{t} \dot{x}_\alpha^T(s) R \dot{x}_\alpha(s) ds d\theta + \\ \qquad \int_{t-\delta}^{t} x_\alpha^T(s) S x_\alpha(s) ds + \int_{-\mu_1}^{\mu_1} \int_{t+\theta-\delta_1}^{t} \dot{x}_\alpha^T(s) R_a \dot{x}_\alpha(s) ds d\theta, \end{array} \tag{3}$$

where $\bar{x}_\alpha(t) = col\{x_\alpha(t), \dot{x}_\alpha(t)\}$, $x_\alpha(t) = x(t)e^{\alpha t}$ and $E = diag\{I, 0_{(2\times 2)}\}$.

Because of the separation principle, one can divide the analysis of the global stabilization into two smaller problems: the observer design and the controller design. The result are recalled hereinafter using an observer/controller.

2.4 Observer design

For a given k and for any $t \in [t_{1,k} + h_{1m}\ ,\ t_{1,k+1} + h_{1m}[$, there exists a k' such that the proposed observer is of the form:

$$\begin{cases} \dot{\hat{x}}(t) = A\hat{x}(t) + Bu(t_{1,k}) - L(y(t_{2,k'}) - \hat{y}(t_{2,k'})), \\ \hat{y}(t) = C\hat{x}(t). \end{cases} \tag{4}$$

The index k' corresponds to the most recent output information that the Master has received. Note that the Master knows the time $t_{1,k}$ and the control $u(t_{1,k})$ (see Section 3.3), which makes this observer realizable.

Using the delay (1) re-writing , one obtains:

$$\begin{cases} \dot{\hat{x}}(t) = A\hat{x}(t) + Bu(t - \delta_1(t)) - L(y(t - \delta_2(t)) - \hat{y}(t - \delta_2(t))), \\ \hat{y}(t) = C\hat{x}(t). \end{cases} \tag{5}$$

with $\delta_1(t) \triangleq t - t_{1,k}$ and $\delta_2(t) \triangleq t - t_{2,k'}$. In other words, the observer is realizable because the times $t_{1,k}$ and $t_{2,k'}$ defining the observer delays are known thanks to the time stamps. The system features lead to $\delta_1(t) \leq h_{1m} + T$ and $\delta_2(t) \leq h_{2m} + T$.

We define the error vector between the estimated state $\hat{x}(t)$ and the present system state $x(t)$ as $e(t) = x(t) - \hat{x}(t)$. From (2) and (5), this error is ruled by:

$$\dot{e}(t) = Ae(t) - LCe(t - \delta_2(t)). \tag{6}$$

Theorem 1. *[13] Suppose that, for some positive scalars α and ε, there exists $n \times n$ matrices $0 < P_1$, P, S, Y_1, Y_2, Z_1, Z_2, Z_3, R, R_a and a matrix W with appropriate dimensions such that the following LMI conditions are satisfied for $j = 1,2$:*

$$\begin{bmatrix} \Psi_2 & \begin{bmatrix} \beta_{2j}WC - Y_1 \\ \varepsilon\beta_{2j}WC - Y_2 \end{bmatrix} & \mu_2\beta_{2j}\begin{bmatrix} WC \\ \varepsilon WC \end{bmatrix} \\ * & -S & 0 \\ * & * & -\mu_2 R_a \end{bmatrix} < 0,$$

$$\begin{bmatrix} R & Y \\ * & Z \end{bmatrix} \geq 0,$$

where β_{2j} are defined by:

$$\begin{array}{l} \beta_{11} = e^{\alpha(\delta_1 - \mu_1)},\ \beta_{12} = e^{\alpha(\delta_1 + \mu_1)}, \\ \beta_{21} = e^{\alpha(\delta_2 - \mu_2)},\ \beta_{22} = e^{\alpha(\delta_2 + \mu_2)}, \end{array} \tag{7}$$

and the matrices Y, Z and Ψ_2 are given by:

$$Y = [Y_1\ Y_2], \quad Z = \begin{bmatrix} Z_1 & Z_2 \\ * & Z_3 \end{bmatrix}, \tag{8}$$

$$\begin{array}{l} \Psi_2^{11} = P^T(A_0 + \alpha I) + (A_0 + \alpha I)^T P + S \\ \qquad + \delta_2 Z_1 + Y_1 + Y_1^T, \\ \Psi_2^{12} = P_1 - P + \varepsilon P^T(A_0 + \alpha I)^T + \delta_2 Z_2 + Y_2, \\ \Psi_2^{22} = -\varepsilon(P + P^T) + \delta_2 Z_3 + 2\mu_2 R_a + \delta_2 R. \end{array}$$

Then, the gain:

$$L = (P^T)^{-1} W, \tag{9}$$

makes the error (6) of observer (5) exponentially converge to the solution $e(t) = 0$, with a decay rate α.

In the following, the solution of the LMI problem corresponding this theorem is written:

$$L = LMI_{obs}(\mu_2, \delta_2, \alpha) \tag{10}$$

2.5 Control design

We first consider a controller $u = Kx$, $i = 1,2$, *i.e.* the ideal situation $e(t) = 0$, $x(t) = \hat{x}(t)$ and:

$$\dot{x}(t) = Ax(t) + BKx(t - \delta_1(t)). \tag{11}$$

Theorem 2. *[13] Suppose that, for some positive numbers α and ε, there exists a positive definite matrix $\bar{P}_1$, matrices of size $n \times n$: $\bar{P}$, $\bar{U}$, $\bar{Z}_1$, $\bar{Z}_2$, $\bar{Z}_3$, $\bar{Y}_1$, $\bar{Y}_2$ similarly to (8) and a $n \times m$ matrix W, such that the following LMI conditions hold:*

$$\Gamma_{3i} = \begin{bmatrix} \Psi_3 & \begin{bmatrix} \beta_{1i}BW - \bar{Y}_1^T \\ \varepsilon\beta_{1i}BW - \bar{Y}_2^T \end{bmatrix} & \mu_1 \begin{bmatrix} \beta_{1i}BW \\ \varepsilon\beta_{1i}BW \end{bmatrix} \\ * & -\bar{S} & 0 \\ * & * & -\mu_1\bar{R}_a \end{bmatrix} < 0,$$
$$\forall i = 1,2,$$
$$\begin{bmatrix} \bar{R} & \bar{Y}_1 & \bar{Y}_2 \\ * & \bar{Z}_1 & \bar{Z}_2 \\ * & * & \bar{Z}_3 \end{bmatrix} \geq 0,$$

where β_{1i}, for $i = 1,2$, are defined by (7) and

$$\begin{aligned} \bar{\Psi}_3^{11} &= (A_0 + \alpha I)\bar{P} + \bar{P}^T(A_0 + \alpha I)^T + \bar{S} \\ &\quad + \delta_1\bar{Z}_1 + \bar{Y}_1 + \bar{Y}_1^T, \\ \bar{\Psi}_3^{12} &= \bar{P}_1 - \bar{P} + \varepsilon\bar{P}^T(A_0 + \alpha I)^T + \delta_1\bar{Z}_2 + \bar{Y}_2, \\ \bar{\Psi}_3^{22} &= -\varepsilon(\bar{P} + \bar{P}^T) + \delta_1\bar{Z}_3 + 2\mu_1\bar{R}_a + \delta_1\bar{R}. \end{aligned}$$

Then, the gain:

$$K = W\bar{P}^{-1}, \tag{12}$$

exponentially stabilizes the system (11) with the decay rate α for all delay $\delta_1(t)$.

In the following, the solution of the LMI problem corresponding this theorem is written:

$$K = LMI_{con}(\mu_1, \delta_1, \alpha) \tag{13}$$

2.6 Global stability of the remote system

The gains K and L have to be computed in such a way they exponentially stabilize the global Master-Slave-Observer system despite the variable delays $\delta_1(t)$ and $\delta_2(t)$. This global system is:

$$\begin{cases} \dot{x}(t) = Ax(t) + BKx(t - \delta_1(t)) + BKe(t - \delta_1(t)), \\ \dot{e}(t) = Ae(t) + LCe(t - \delta_2(t)). \end{cases} \tag{14}$$

A separation principle is then applied to the previous system. Then if L and K are chosen with respect to sections 2.4 and 2.5 respectively, the global system is exponentially stable with the lowest decay rate.

3 Architecture of the global control system

3.1 Features of the remote system

The remote system is based on Master-Slave structure. In order to simplify the work of the Slave PC, the control and observation complexity is concentrated on

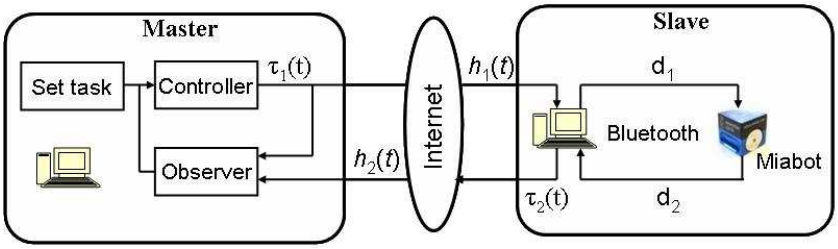

Fig. 3. Structure of the global system

the Master. The main features of the system refer to Fig.3. In the system, the robot Miabot of the company Merlin Systems Corp. Ltd together with a PC serves as the Slave. The Miabot works as an accessory device which communicates with the PC by the port of Bluetooth, so we cannot use buffer strategy directly on the robot. Because there is no time information included in the command of Miabot, we do not know when the control is applied. That means, we cannot use time stamp on Miabot. To simplify our problem and apply the theory of [13], we treat the time-delay of Bluetooth between the PC and Miabot as a constant one. We add the delays d_1, d_2 into the respectively variable delays $h_1(t)$, $h_2(t)$.

The transmission protocol UDP is applied to communicate the data between Master and Slave. In order to know the instant of data-sent, time stamps are added to the data packets. The data structure of list served as buffers is introduced for the program to search for the data of the right instance. In all the lists, the control data are restored in the decreasing order of its sending time. That is to say, the most recent sent data is always at the beginning of the list.

3.2 Synchronization of the time clocks

To synchronize the different time in the two PCs, we can add GPS into the system [13], but this increases the cost and it is not flexible. Another way is to use a certain protocol such as NTP (Network Time Protocol) [20]. Due to different time clock of PCs, we have to make synchronization from time to time. So, our solution is to directly adapt the strategy of NTP in the program to calculate the time differences.

As showed in Fig. 4, here k is the sequential number of the packets sending from the Master and k' is the number sent back. $h_1(k)$ and $h_2(k')$ refer to the respective delays of the communication on Internet. To simplify the problem, we assume that $h_1(k) = h_2(k')$. If we define the time clock difference between the M/S as follows (t_s,t_m are the respectively time of the Slave and the Master):

$$\theta = t_m - t_s; \tag{15}$$

Then, we can calculate the time difference between the two PCs using the following equations ($\triangle$ is the process time for Slave):

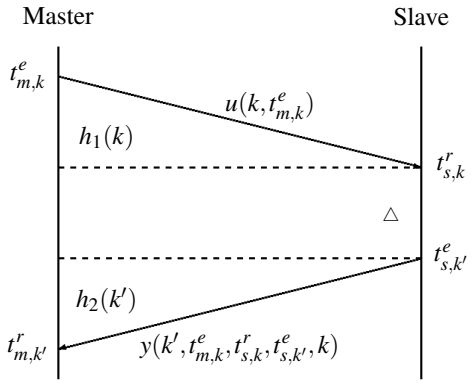

Fig. 4. Packets communication between the M/S

$$\theta(k,k') = (t^r_{m,k'} - t^e_{s,k'} + t^e_{m,k} - t^r_{s,k})/2; \tag{16}$$

$$h_1(k) = h_2(k') = (t^r_{s,k} - t^e_{m,k} + t^r_{m,k'} - t^e_{s,k'})/2; \tag{17}$$

That is to say, every time the Master receives a packet, the time clock difference between the M/S and the time delay of the Internet can also be measured. The values of θ and h_m are comprised in the control packets, so whenever the Slave receives a control, it can calculate the "target" time for applying the control.

3.3 Transmission and receipt of the control data

The k^{th} data sent by the Master to Slave includes the control $u(t^e_{m,k})$ together with the time stamp when the packet is sent out. At time $t^r_{s,k}$, when the Slave receives the data it can calculate the time delay because of the time stamp. If the delay equals $h_{1m} - d_1$, the Slave should apply immediately the command.

The control u, sent out by the Master at time $t^e_{m,k}$, is received by the Slave at time $t^r_{s,k} > t^e_{m,k} - \theta$. It will be injected in the Slave input only at the pre-defined “target time": $t^{target}_{s,k} = t^{target}_{m,k} - \theta = t^e_{m,k} + h_{1m} - \theta$. The corresponding waiting time h_{1m} is depicted on Fig. 5. This is realistic because the transmission delay is bounded by a known value h_{1m}. By this way, the Master knows the time when this control $u(t^e_{m,k})$ will be injected at the Slave input.

3.4 The structure of the Master

In order to implement the model for the remote control system, four-thread program is designed to fulfill the functions of Controller and Observer in Fig.3.

These four threads are concurrently working as showed in Fig.6. There are two buffers, *list_U* and *list_X* which respectively keep the data sent out from the Master and the data of the estimated state of the Slave. The most recent calculated data is

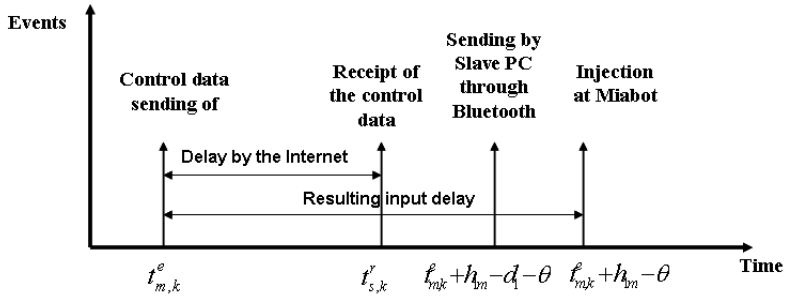

Fig. 5. Control data processing

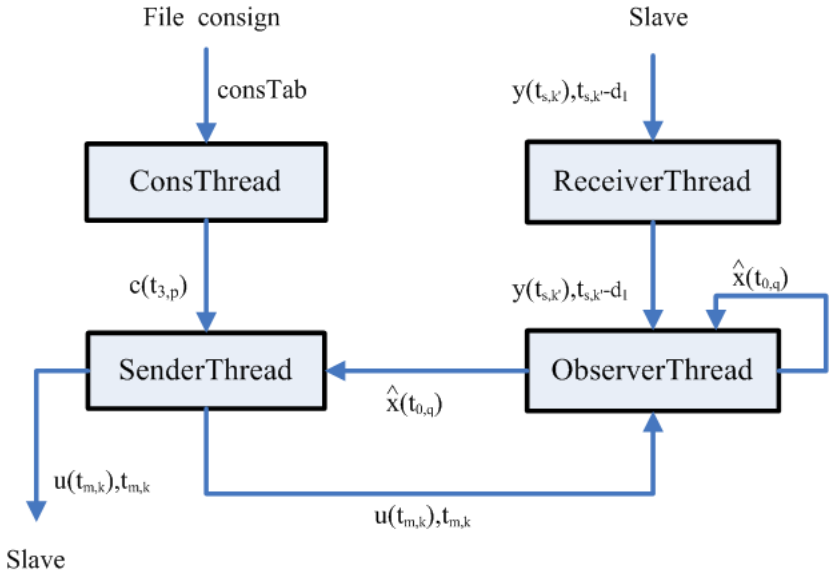

Fig. 6. Structure of the Master

inserted at the beginning of the lists so that it is easier to find the right data we need and delete the obsolete ones.

Here we consider the following notations. p is the index of the different commands given by the ConsThread. k is the index of the control u given by SenderThread. q is the index of estimation state given by the ObserverThread and the k' is the index of the measure given by the ReceiverThread.

(a) ConsThread is a periodic thread which gets the tasks (it is the position where the user wants the motor to arrive) from a file given by the user. In this way, the user can freely change the task. The time period T_3 for this thread to work continuously is

set before the system begins to work whose value should be greater than the response time of the Slave.

(b) SenderThread is also a periodic one which first gets the task ($c(t_{3,p})$) designed by the user. Considering the mechanic character of the Miabot, we choose $0.1sec$ as the time period. Then it calculates the control data for sending out to the Slave.

The most recent $\hat{x}(t_{0,q})$ can be found at the beginning of the *list_X* ; then, the data of command together with the system time is sent out to the slave through a socket. While, at the same time it is inserted into the *list_U* for the ObserverThread to use.

(c) ReceiverThread is a event-driven thread. As there is data arrived from the slave, it first check whether there is packet loss. The maximum number of packet which can be lost without implies instability of the system is an open issue of our researches. Then according to the time stamp, the time clock difference and the time-delay are calculated, meanwhile, the most recent data is sent to the thread of ObserverThread.

(d) ObserverThread is the most important part of the program. It mainly serves as the Observer in the system model. The main task is to estimate the *present* position and speed of the motor. In order to estimate the continuous state of the Miabot, the time period of this thread should be small enough. In our program, we choose $0.01sec$. As we can see from the result of the experiment (Section 3.6), this value is suitable. To accomplish the function of the Observer, it is needed to find out the command u which has been applied to the slave system and the estimated motor position at the time when the information is sent out from the slave.

As it is illustrated in Fig.7, in order to determine $\hat{y}(t_{2,k'})$, it is necessary to find in the *list_X* the closer state estimation $\hat{x}$ with regard to the date $t_{s,k'} - d_1$. And we can get the control data u in the *list_U* with the time stamp of time h_{1m} before. So, according to the equation (4), the estimated state can be obtained. As we can see

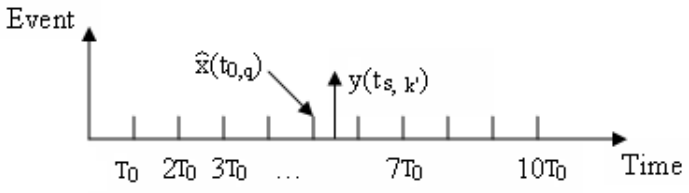

Fig. 7. Packet Sequences

from Fig.7, in order to find the state $\hat{x}(t_{0,q})$ at the time nearly to $t_{s,k'} - d_1$, the time period of this thread should be small enough.

3.5 The structure of the Slave

The Slave does not need power computation abilities, because it just needs to communicate with the Master and the Miabot. As we can see from Fig.8, this pro-

gram is divided into two threads: ReceiveThread and SendThread. As we need to apply the control data with the time delay of h_{1m} after the time stamp, a *list_Y* is used to contain the control data temporarily, in which all the nodes are sorted in the order according to the its' time stamp. That means the most recent control is inserted at the end of the list.

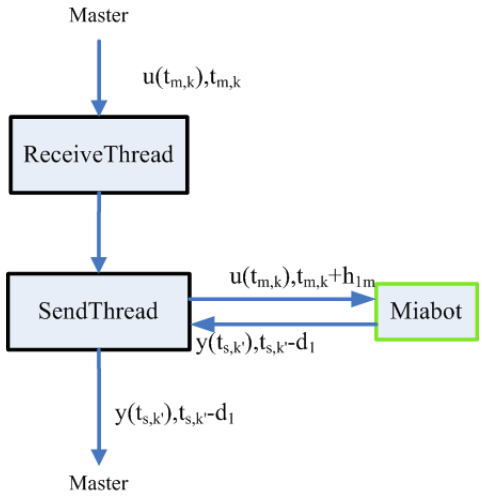

Fig. 8. Structure of the Slave

(a) ReceiveThread is an event-driven thread which is activated by the control data arrived from the Master. The control data is inserted into the proper position of the list *list_Y* according to its time stamp. If the time stamp is before the oldest data of the list, that means there is disorder of the packets through Internet, then the data is discarded. If there are several packets lost, as we have a high frequency of the Master's SenderThread, it does not affect the stability of the system.

(b) SendThread is used to apply the control to the Miabot as well as to get its real position, and then send the data back to the Master. Firstly, the thread takes the packet at the beginning of *list_Y*. Thanks to the value of the difference time clock between Master and Slave in the packet, we can calculate the "target" time to apply the control. Then the control is sent to the Miabot by the port of Bluetooth at the right time when the Miabot sends back the measure $y(t^e_{s,k'})$ of its position. This value is sent to the Master with the time stamp $t_{s,k'} - d_1$ while $t_{s,k'}$ is the reception time by the Slave PC.

3.6 Experimental study

After identification of the Miabot, we get the following model:

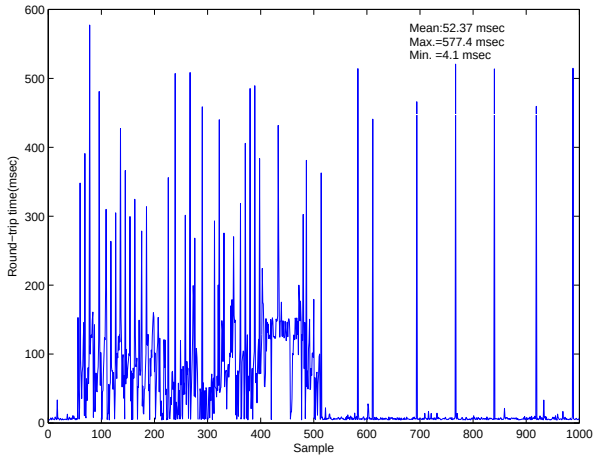

Fig. 9. The RTT between the two PCs by Internet(40km away)

$$\begin{cases} \dot{x}(t) = \begin{bmatrix} 0 & 1 \\ 0 & -10 \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ 0.014 \end{bmatrix} u(t - \delta_1(t)) \\ y(t) = \begin{bmatrix} 1 & 0 \end{bmatrix} x(t). \end{cases} \tag{18}$$

We have continuously tested the RTT (Round-trip-time) between the two PCs by the protocol ICMP (Internet Control Message) as showed in Fig.9. From these tests, considering also the Bluetooth transmission delays and the sampling delays, we take the value $\delta_1 = \delta_2 = 0.4sec$ and $\mu_1 = \mu_2 = 0.1sec$. If there are once or a few times that the time-delays are bigger than $0.5sec$, we treat the packets as being lost. The gains K and L have to be computed in such a way they exponentially stabilize the global Master-Slave-Observer system despite the variable delays $\delta_1(t)$ and $\delta_2(t)$. According to [13], we get $\alpha = 0.96$ when the gain L is chosen as:

$$L = \begin{bmatrix} -1.4 \\ -0.13 \end{bmatrix}. \tag{19}$$

The gain K is as following:

$$K = \begin{bmatrix} -702 & -70 \end{bmatrix} \tag{20}$$

The experiment is done on two computers separated about 40 kilometers away. The Master program runs on the remote computer with an advanced computing capability, the slave program on the local one. We get the result showed in the Fig. 10, in which the chain line represents the set values; the solid lines represent the robot's estimated state, the position and the speed; the dotted line correspond to the real position. Fig.11 represents the sampled control data send to Slave.

Note that all the data in the figure are obtained from the Master, so the data of the real position of the Miabot (dotted line) lags behind the estimated one. This

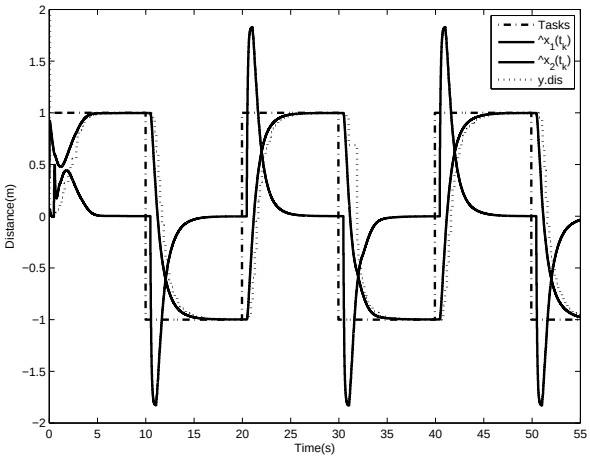

Fig. 10. Results of remote experiment

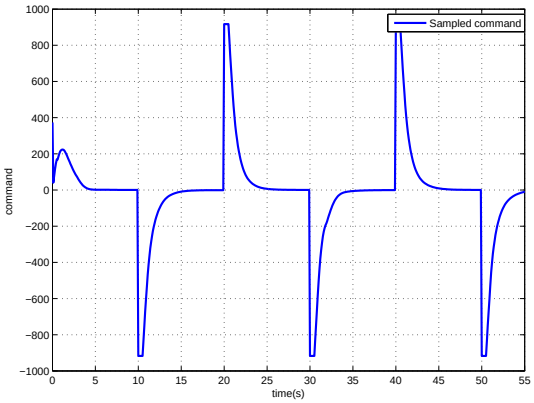

Fig. 11. The corresponding Slave control of centralization experiment

illustrates the fact that, despite the time delays of Internet and Bluetooth, the Master estimates a prediction of the Slave's state.

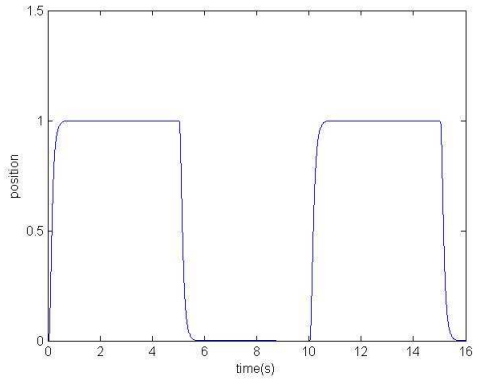

Fig. 12. $h_m = 0.05s, \alpha = 8.74$

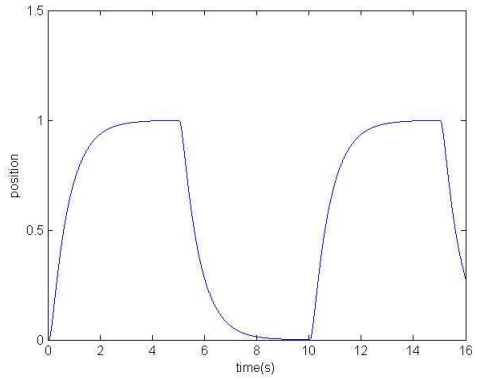

Fig. 13. $h_m = 0.5s, \alpha = 0.96$

h_m	0.02	0.04	0.06	0.08	0.1	0.2	0.3	0.4	0.5
α	9.99	9.99	7.54	5.94	4.91	2.59	1.71	1.25	0.96

Fig. 14. The relation between h_m (sec) and α

4 Performance enhancement by a gain scheduling strategy

4.1 Effects of time-delay on the performance and the system stability

The time-delay of the Internet varies a lot especially between the rush hour and idle time period (Fig. 9). In order to guarantee the exponential stabilization, we have to choose the maximum time-delay, whereas most of the time, the time-delay is much

smaller. In other words, the performance is decreased. Comparing the two simulation results of Matlab, Fig. 12 and Fig. 13, we can see that when the h_m is smaller, the value of α is larger and the task is more quickly attained. In the table Fig. 14, we list several corresponding values of h_m and α. It is clearly that increasing h_m means decreasing the performance of the system.

In order to enhance the performance and make the system adaptable to the changeable time-delay of the Internet, we have designed two controllers corresponding with two bounds of time-delay. The controller switches on the function of time-delay. The switching signal is given by $\sigma(t) = \gamma(t - \xi)$, where ξ is the time-delay of the signal due to the Internet and calculation. In order to guarantee the *uniform* exponential stability, our solution is to find a common Lyapunov function for both closed loops [23]. Of course, for greater delay values, the performance cannot be guaranteed anymore and an alternative solution has to be considered. In our system, we give a command for the robot to stop until the communication comes back to a sufficient quality.

4.2 *Uniform* stability with gain scheduling

In order to reach higher value for the exponential convergence, one proposes switching controller and observer gains. The switching signals $\sigma_1(t)$ and $\sigma_2(t)$ chosen are function of the time delays $\delta_1(t)$ and $\delta_2(t)$. For sake of simplicity, they can only take two values:

$$\sigma_i(t) = j, if\, \delta_i(t) \in [h_{Min}^{ij}, h_{Max}^{ij}[, i, j = 1,2 \tag{21}$$

Consider every zone of time-delay, we have to compute the gains K_1, K_2 and L_1, L_2 in such a way that they exponentially stabilize the global Master-Slave-Observer system despite the variable delays $\delta_1(t)$ and $\delta_2(t)$. This global system is:

$$\begin{cases} \dot{x}(t) = & Ax(t) + BK_{\sigma_1(t)}x(t - \delta_1(t)) + BK_{\sigma_1(t)}e(t - \delta_1(t)), \\ \dot{e}(t) = & Ae(t) + L_{\sigma_2(t)}Ce(t - \delta_2(t)), \\ \sigma_1(t) = & \gamma_1(t - \delta_1(t) - \delta_2(t)), \\ \sigma_2(t) = & \gamma_2(t - \delta_2(t)). \end{cases} \tag{22}$$

$\gamma_i(t)$ are the detection functions of $\delta_i(t)$. These functions are delayed because the Master needs to receive the last packet to calculate the delays. So each gain is activated for a certain period ($\delta_1(t) + \delta_2(t)$) or $\delta_2(t)$)).

Each gain is computed using

$K_j = LMI_{con}((h_{Max}^{1j} - h_{Min}^{1j})/2, (h_{Max}^{1j} + h_{Min}^{1j})/2, \alpha_{con})$

$L_j = LMI_{obs}((h_{Max}^{2j} - h_{Min}^{2j})/2, (h_{Max}^{2j} + h_{Min}^{2j})/2, \alpha_{obs})$.

A sufficient condition to prove the uniform stability of the switching closed loop is to find a common Lyapunov-Krasovskii functional for all gains. This functional has to take into account all admissible delays i.e. $\forall \delta_i(t) \in [h_{Min}^{i1}, h_{Max}^{i2}[$. For each gain Fig. 15 shows the regions where the exponential stability is proven.

The following theorems give sufficient conditions to prove the uniform stability of the switching closed loop.

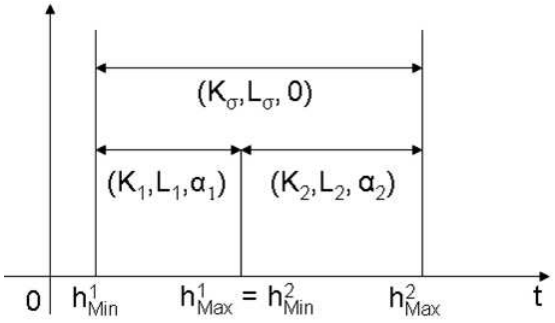

Fig. 15. Uniform stability of the system

Theorem 3. *Suppose that, for a given switching observer gains $L_{\sigma_2(t)}$, for some positive scalars α and ε, there exists $n \times n$ matrices $0 < P_1$, P, S, Y_1, Y_2, Z_1, Z_2, Z_3, R, R_a such that the LMI conditions 1 and the following ones hold for $i = 1,2$, $j = 1,2$*

$$\begin{bmatrix} \Psi_2 & \begin{bmatrix} \beta_{2j}P^TL_iC - Y_1 \\ \varepsilon\beta_{2j}P^TL_iC - Y_2 \end{bmatrix} & \mu_2\beta_{2j}\begin{bmatrix} P^TL_iC \\ \varepsilon P^TL_iC \end{bmatrix} \\ * & -S & 0 \\ * & * & -\mu_2 R_a \end{bmatrix} < 0,$$

where the matrices Ψ_2 and β_{2j} are the same as in theorem 1.

Then the error of observer exponentially converge to the solution $e(t) = 0$, with a decay rate α.

Proof: Consider the Lyapunov-Krasowskii functional (3). Following the same proof as in [13], one gets the following sufficient conditions for $j = 1,2$:

$$\begin{bmatrix} \Psi_2 & \begin{bmatrix} \beta_{2j}P^TL_{\sigma_2(t)}C - Y_1 \\ \varepsilon\beta_{2j}P^TL_{\sigma_2(t)}C - Y_2 \end{bmatrix} & \mu_2\beta_{2j}\begin{bmatrix} P^TL_{\sigma_2(t)}C \\ \varepsilon P^TL_{\sigma_2(t)}C \end{bmatrix} \\ * & -S & 0 \\ * & * & -\mu_2 R_a \end{bmatrix} < 0,$$

Then by convexity, one obtains the conditions of the previous theorem.

Theorem 4. *Suppose that, for a given switching state feedback $K_{\sigma_1(t)}$, for some positive numbers α and ε, there exists a positive definite matrix $\bar{P}_1$, matrices of size $n \times n$: $\bar{P}$, $\bar{U}$, $\bar{Z}_1$, $\bar{Z}_2$, $\bar{Z}_3$, $\bar{Y}_1$, $\bar{Y}_2$ similarly to (8), such that LMI conditions 1 and the following ones hold for $i = 1,2$, $j = 1,2$*

$$\begin{bmatrix} \Psi_3 & \begin{bmatrix} \beta_{1i}BK_j\bar{P} - \bar{Y}_1^T \\ \varepsilon\beta_{1i}BK_j\bar{P} - \bar{Y}_2^T \end{bmatrix} & \mu_1\begin{bmatrix} \beta_{1i}BK_j\bar{P} \\ \varepsilon\beta_{1i}BK_j\bar{P} \end{bmatrix} \\ * & -\bar{S} & 0 \\ * & * & -\mu_1\bar{R}_a \end{bmatrix} < 0,$$

where the matrices Ψ_3 and β_{1j} are the same as in theorem 1. Then the closed loop is exponentially stable with the decay rate α for all delay $\delta_1(t)$.

Proof: Same as in the observer case.

4.3 Gain scheduling experiments

Considering the initial approach, i.e. without switching gains, the maximum exponential convergence obtained is $\alpha_{control} = \alpha_{observer} = 0.96$.

Consider two zones of delay with $\delta_1^1 = \delta_2^1 = 0.04sec.$,$\mu_1^1 = \mu_2^1 = 0.04sec$,and $\delta_1^2 = \delta_2^2 = 0.29sec.$, $\mu_1^2 = \mu_2^2 = 0.21sec$. It means that the gains switch when the delay crosses the value of $0.08sec$. According to *Theorem (3) and (4)*, the maximum exponential convergence ensuring the global stability are: $\alpha_{control1} = 2.1$, $\alpha_{observer1} = 2.2$, and $\alpha_{control2} = \alpha_{obsever2} = 1$.

Note that because the global stability is checked after the computation of the gains, these values are not optimal. To get optimal value, it is needed to be able to find the control/observer gains ensuring the exponential convergence and the global stability at the same time with a LMI problem.

The gains K_i and L_i $(i = 1,2)$ are:

$$L_1 = \begin{bmatrix} -3.01 \\ -0.77 \end{bmatrix}, K_1 = \begin{bmatrix} -1659 & -260 \end{bmatrix}.$$

$$L_2 = \begin{bmatrix} -1.4 \\ -0.16 \end{bmatrix}, K_2 = \begin{bmatrix} -1015 & -100 \end{bmatrix}.$$

4.4 Result of remote experiment

The experiment is done in the same situation as mentioned in the proceeding parts. The result is shown in Fig. 16, in which the chain line represents the set values; the solid lines represent respectively the robot's estimated position and speed; the dotted line corresponds to the real position of the Miabot. Fig.17 is the corresponding variable time-delays, which comprises the time-delay of sampling and communication of Bluetooth (we consider it as constant time-delay, here we take the value of 40ms). In Fig.18, the solid line represents the sampled control sent to Slave, and the chain line and dotted one represent the command for the zone one and two respectively. Fig.19 shows the time point when the system switches enter the values of the two zones.

Because the maximal speed of the Miabot is $3.5m/sec$ [24], the command value corresponding is 2000 (in open loop). But to guarantee the linear character of the Miabot, we make the command not greater than 1000 and the speed $1.7m/sec$. The controller gains are those of the last section. Despite of their high value, one can notice that the control signal (Fig.18) to not exceed the limit value as well as the speed (Fig. 16 solid line) which validates the linearity assumption.

On Fig. 16, one can notice three kinds of step response. The first one corresponds to the case where the control switches a lot during the response. In that case, only the global stability is guaranteed. During the second step, only the second zone is active, i.e. only the gains K_2 and L_2 are active ($\alpha = 1$). In this case, some performances are guaranteed. In the last kind of response, only the first zone is active because the delays are small. In that case, the performances are better ($\alpha = 2.1$): the response time is smaller and the damping is greater.

As it is clearly shown in Fig. 16, the global stability of the closed loop is maintained despite that some assumptions made are not satisfied. On the bluetooth, it was considered constant whereas in reality it varies (the minimum delay recorded os less than 40*ms*). And the other one is on the synchronization, symmetric delays were needed and in the experiment it was clearly not the case.

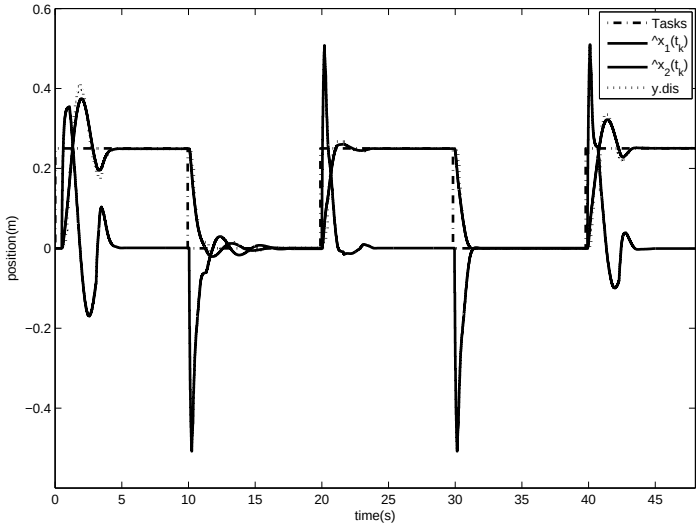

Fig. 16. Results of remote experiment

5 Conclusion

In addition to some fundamental results, an experimental platform has been developed to illustrate the results of the network-based control theory. This platform is able to control a slave through a network and joins skills in automatic control, computer science and networks.

The experimental results confirm the theory: 1) The exponential stability is obtained in both the time-delay zones and the *uniform* stability is guaranteed. 2) The experimental performances are showed to be better when considering two zones of time delay instead of one.

Considering the variation of time-delays, more than two-zone-switching signals can be selected in order to enhance the performance of the global system. The LMI conditions, in that case, have an increased size and are straightforwardly inspired from *Theorem (3)* and *(4)* are not investigated here.

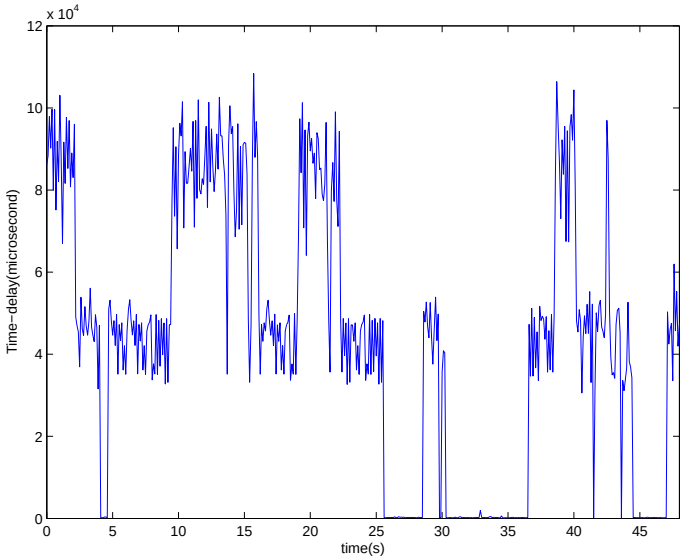

Fig. 17. The corresponding variable time-delays

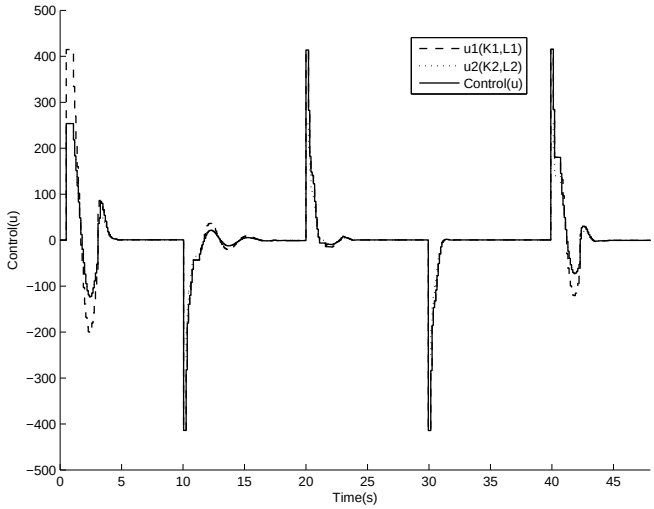

Fig. 18. The corresponding Slave control

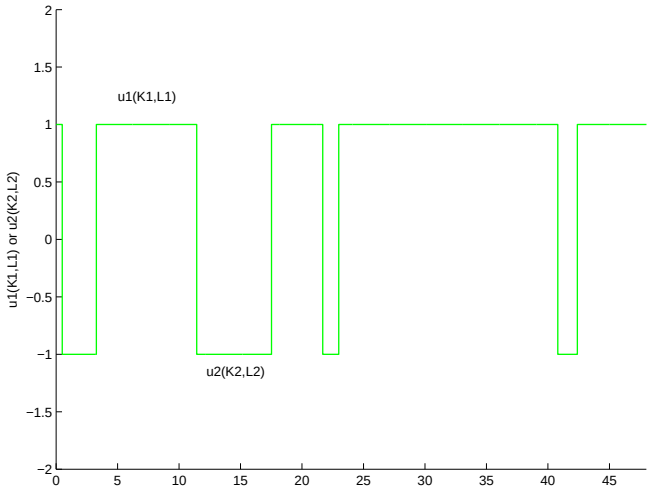

Fig. 19. The switching plan

A way to improve the presented results is to propose a "one shot algorithm" which allows finding the optimal gains in term of exponential convergence. Another trend is to investigate a solution without the input buffer. Without buffer, the input delay will be smaller ensuring more performances and the slave will need less memory to run [25].

A last perspective is to consider the improvement of the network communication by, for example, developing dedicated protocols which minimize the time delays and enhance the clock synchronization.

References

1. Yu M., Wang L., and Chu T., "An LMI approach to network control systems with data packet dropout and transmission delays," *MTNS '04 Proc. of Mathematical Theory Networks and Systems, Leuven, Belgium*, 2004.
2. Richard J.P. and Divoux T., *Systèmes commandés en réseau*. Hermes-Lavoisier, IC2, Systèmes Automatisés, 2007.
3. Georges J.-P., Divoux T., and Rondeau E., "Confronting the performances of a switched ethernet network with industrial constraints by using the network calculus," *International Journal of Communication Systems(IJCS)*, vol. 18, no. 9, pp. 877–903, 2005.
4. Chiasson J. and Loiseau J.J., *Applications of time delay systems*. Springer, 2007, vol. 352.
5. Hale J. and Lunel S., "Introduction to functional differential equations," *Springer*, 1993.
6. Kolmanovskii V. and Myshkis A., *Applied theory of functional differential equations*. Dordrecht:Kluwer, 1999.

7. Niculescu S.-I., *Delay effects on stability: a robust control approach*. Springer, 2001, vol. 269.
8. Richard J.P., "Time delay systems: an overview of some recent advances and open problems," *Automatica*, vol. 39, pp. 1667–1694, 2003.
9. Almutairi N.B., Chow M.Y., and Tipsuwan Y., "Network-based controlled dc motor with fuzzy compensation," *Proceedings of the annual conference on industrial electronics society, Denver*, pp. 1844–1849, 2001.
10. Huang J.Q. and Lewis F.L., "Neural-network predictive control for nonlinear dynamic systems with time delays," *IEEE Transactions on Neural Networks*, vol. 14, no. 2, pp. 377–389, 2003.
11. Leleve A., Fraisse P., and Dauchez P., "Telerobotics over IP networks: Towards a low-level real-time architecture," *IROS'01 International conference on intelligent robots and systems,Maui,Hawaii*, October 2001.
12. Witrant E., Canudas-De-Wit C., and Georges D., "Remote stabilization via communication networks with a distributed control law," *IEEE Transactions on Automatic control*, 2007.
13. Seuret A., Michaut F., Richard J.P., and Divoux T., "Networked control using gps synchronization," *Proc. of ACC06, American Control Conf., Mineapolis, USA*, June 2006.
14. Sandoval-Rodriguez R., T.Abdallah C., N.Jerez H., Lopez-Hurtado I., Martinez-Palafox O., and Lee D., *Applications of time delay systems*. Springer, 2007, vol. 352, ch. Networked Control Systems: Algorithms and Experiments, pp. 37–56.
15. Azorin J.M., Reinoso O., Sabater J.M., Neco R.P., and Aracil R., "Dynamic analysis for a teleoperation system with time delay," *Conf. on Control Applications*, 6 2003.
16. Fattouh A. and Sename O., "H$^\infty$-based impedance control of teleoperation systems with time delay," *4th Workshop on Time Delay Systems*, 9 2003.
17. Niemeyer G. and Slotine J.-J., "Towards force-reflecting teleoperation over the internet," *IEEE Int. Con. on Robotics & Automation*, 1998.
18. Chen Z., Liu L., and Yin X., "Networked control system with network time-delay compensation," *Industry Applications Conference, Fourtieth IAS Annual Meeting*, vol. 4, pp. 2435 – 2440, 10 2005.
19. Seuret A., Dambrine M., and Richard J.P., "Robust exponential stabilization for systems with time-varying delays," *Proc. of TDS04, 5th IFAC Workshop on Time Delay Systems, Leuven, Belgium*, September 2004.
20. Mills D.L., "Improved algorithms for synchronizing computer network clocks," *IEEE/ACM Transactions On Networking*, vol. 3, no. 3, pp. 245–254, June 1995.
21. Seuret A., Fridman E., and Richard J.P., "Sampled-data exponential stabilization of neutral systems with input and state delays," *Proc. of IEEE MED 2005, 13th Mediterranean Conference on Control and Automation, Cyprus*, 2005.
22. Fridman E., Seuret A., and Richard J.P., "Robust sampled-data stabilization of linear systems: An input delay approach," *Automatica*, vol. 40(8), pp. 1441–1446, 2004.
23. Liberzon D., *Switching in Systems and Control*, T. Basar, Ed. Birkhäuser, 2003.
24. Merlin Systems Corporation Ltd, *Miabot PRO BT v2 User Manual*, rev. 1.3 ed., 2004.
25. Seuret A. and Richard J.P., "Control of a remote system over network including delays and packet dropout," *IFAC World Congress, Seoul, Korea*, 7 2008.